

Paralelno programiranje

Drugi popravni kolokvijum

11.6.2015.

1. Organizovano je višemilionsko takmičenje u čuvenoj igri „**Papir-kamen-makaze**“ i to u cilju konačne odluke šta je najjače, odnosno papir, kamen ili makaze. Da bi se to pokazalo potrebno je odigrati veliki broj partija 1 na 1. S tim u vezi, organizatori su rasporedili igrače na veliku tablu, i to u m vrsta sa po n takmičara. Igra se odvija u k krugova. U svakom krugu, svaki igrač izabere papir, kamen ili makaze, a potom se takmiči sa svim svojim susedima (najviše 8, najmanje 3). Svako za sebe beleži broj pobeda i poraza. Na kraju svako ko ima pozitivan skor dobija prikladan poklon. Organizatori zapisuju ukupan broj pobeda koje su ostvarili igrači koji su izabrali papir, kamen ili makaze. Nakon k krugova, pada odluka šta je najbolje izabrati!

Napisati paralelni program koji simulira ovu manifestaciju (**domen dekompozicija**)! Igrače predstaviti jednim celim brojem koji predstavlja trenutno izabrano „oružje“ (0-papir, 1-kamen, 2-makaze). Kreirati strukturu **Stat**, koja sadrži tri atributa: *oruzje (int)*, *br_pobeda (int)*, *br_poraza (int)*. Takođe kreirati i operaciju redukcije *SumStat* koja koja sumira strukturu *Stat* i po broju pobeda i po broju poraza. Rezultati simulacije treba da budu:

- Statistika koliko je do svakog koraka bilo pobeda igrača koji su izabrali papir, kamen ili makaze. Koristiti strukturu *Stat* i operaciju redukcije *SumStat*.
- Globalnu poziciju igrača sa najviše pobeda. Ukoliko ima više njih sa istim brojem pobeda, odštampati pozicije svih takvih igrača.

Zakljucno sa krugom	Broj pobeda		
	Papir	Kamen	Makaze
1	10	20	30
2	30	50	40
3	60	65	55
...
k	615	645	660

Fig 1. Primer statistike

2. **BoostMPI**. Kreirati dve klase:

Stat – klasa koja sardži četiri javna podatka: *id (int)*, *br_poena (int)*, *br_skokova (int)*, *br_asistencija (int)*. Sadrži i konstruktor klase koji prima ta četiri podatka i kreira objekat.

Kosarkas – Klasa treba da poseduje sledeće attribute i metode:

- *id* – identifikacioni broj košarkasa, tipa *int*
- *ime* – ime košarkaša, tipa *string*
- *br_poena* – broj postignutih poena, tipa *int*
- *br_asistencija* – broj asistencija, tipa *int*

- *br_skokova* – broj skokova, tipa *int*
- *br_utakmica* – broj odigranih utakmica, tipa *int*
- *Konstruktor* – default
- *Konstruktor* – za dati identifikacioni broj i ime košarkaša kreira objekat
- *AddStat* – metoda koja za dati objekat klase *Stat* povećava brojeve poena, asistencija i skokova košarkaša
- *IsPlayer* – koja za zadati ceo broj proverava da li je to košarkaš sa tim identifikacionim brojem
- *Operator >* - ispituje koji je košarkaš bolji po *mvp_rate* – u. *Mvp_rate* se računa po sledećoj formuli:
$$mvp = \frac{br_poena \cdot 2 + br_asistencija \cdot 3 + br_skokova \cdot 3}{br_utakmica}$$

Napisati paralelan program koji prvo učitava podatke o svim košarkašima, i raspoređuje ih po svim procesima podjednako (broj košarkaša je deljiv brojem procesa). Potom učitava statističke podatke koji u svakom redu sadrže identifikacioni broj igrača, i njegov broj poena, asistencija i skokova na jednoj utakmici. Obraditi ulazne podatke, a potom odštampati najboljeg igrača (MVP).